

周报（2014.0.6.02-2014.06.08）

1、本周工作：

周一：

准备周三的主题报告，阅读曲老师的 07 年的一篇关于香港空气污染数据可视化的文章，重点关注文章中的可视设计。这篇文章采用的基本可视化方法是平行坐标、极坐标和完全加权图。完全加权图用于揭示所有维度之间的相关性，确定平行坐标中轴的顺序。对于平行坐标主要的改进是针对风向这种表示方向的维度用 S 型坐标轴表示。对极坐标改进为圆形的 pixel bar chart。阅读的第二篇文章是曲老师的一篇关于文本数据可视化的文章。关注点也主要在可视设计上。主要抽取文本中的话题演变趋势、重要事件、关键词的关联。用流的方式可视化主题，对于重要事件，分别用小的 glyph 表示事件的开始、结束、split 和 merge。每个关键词用一根线表示，线之间的相互交织表示这些关键词的共同出现。

周二：

为了准备主题报告，阅读了 2013infovis 的一篇关于 glyph 设计的文章 Taxonomy-Based Glyph Design—with a Case Study on Visualizing Workflows of Biological Experiments。这篇文章首先提出了一个比较系统的过程来进行 glyph design，比如一个 glyph 的基本可视化区域的划分、针对这些不同的区域，常见的可视化方案是否适用等。然后通过文章前面部分提出的设计过程，将生物实验工作流中的一个节点设计成合适的 glyph。也阅读了 2012infovis 中的 Graphical Overlays: Using Layered Elements to Aid Chart Reading。不过读完之后发现与可视设计的关系不是太大。

周三：

和芯芯讨论本周的主要工作：建立 local 查询与 global 的查询、local ontology 到 global ontology 的映射。

目前的方案是针对各个 local 的数据源，我们手动（或用程序）建立 local ontology。而不同的 local ontology 实际表示的内容可能有相同的地方，正是这些相同的地方，将不同的数据源中查询的结果整合在了一起。之前实现的方法是手动地将不同的 local ontology 中的内容整合成一个 global ontology。今天的工作是

将 local ontology 用程序整合成 global ontology。

具体的实现方法：之前尝试过解决这个问题，查询到可以用 jena 的 OntModel 中的 add 方法实现。不过当时发现用这个方法时候，会把 owl 中的默认的基本数据类型、基本关系类型等内容产生出来，所以刚开始放弃了这种方法。在查询过程中发现 Brain 这个库（也是 java 的一个关于 ontology 的包，主页是 <http://loopasam.github.io/Brain/>）中也提供了各种基本的 ontology 处理方法，不过这个库产生的 owl 文件格式是 OWL 2 EL 的格式，与我们之前的默认格式不符。又重新转到了用 jena 的 add 方法。后面的时间采用这个方法将 local ontology 合并的一个 global ontology。现在无法进行可视化（ontology 中有各种系统默认的数据），打算第二天处理这个问题。

周四：

目前产生的 owl 文件无法用之前的代码可视化，我也不确定自己写的这部分是否正确。不过界面的部分是解聪写的，而他不在学校。所以尝试看他调试他写的那部分的代码，也与他在网上进行了沟通。解决方法是找到最初的读 owl 文件的部分，在构建 owl 图，在图中添加节点和边的时候添加了判断条件，不会将系统默认的部分产生出来。不过目前还有一个小问题：图中无法区分 ontology 和 instance。这点打算等哈哈来了再请他解决，图的这部分代码是他写的，他比较熟悉。

周五：

添加了一些基本的接口，调整了部分代码的结构。具体的包括：将 local ontology 和 global ontology 的构建剥离出来，单独用两个类实现。添加 local ontology 到 global ontology 的映射文件，就是实现了一个基本的映射类。目前仅仅是一个接口，由于当前的 ontology 都比较简单，还不需要映射。将 owl 文件中的 namespace 单独写到一个全局类中。

下周工作：

1. 将 local 的查询结果先产生 local 的 owl，在整合到 global 的 owl。目前仅仅是

直接将 local 的查询结果产生 global 的 owl。

2. 大家讨论制定下一步的具体的工作。